# (de)Coding the Studio Method to Teach the Design of Human-Computer Interaction

## INTRODUCTION

This paper reports on the beginning of a three-year project funded by the National Science Foundation (NSF) to apply the studio method to teach computer science students principles of user interface design. The grant spans three universities and four disciplines, with a research team of faculty drawn from computer science, education, architecture and industrial design. The goal of this project is to leverage knowledge about design education from architecture and industrial design to develop new educational models and materials for the design of software-intensive systems, specifically in the area of Human Computer Interaction (HCI).

Computer Science *is*, in many ways, a design discipline. For example, application areas such as graphics and visual programming, artificial intelligence, information systems, and human computer interaction, require the design of algorithms, interfaces, interactions, programs, specifications, simulations, and/or systems. A few innovative computer science programs have implemented the studio method. In these cases the logistics and procedures involved have been well documented, but little is known about *which* components of the studio experience are critical to successful outcomes. Thus, our aim is to determine through qualitative research an elemental set of interactions that contribute to studio learning. Further, we will identify effective ways of applying these lessons to teaching design in human computer interaction.

In this paper, we review the nature of design, the use of the studio method in teaching, both in schools of design and the wider university, and relate our initial discussions on transferring the studio to computer science. The nature of a "hybrid studio" in HCI is demonstrated through describing a course that we are currently examining to gather baseline data for our research. We conclude with a set of questions to take back to architectural design education.

## DESIGN

The creative process that synthesizes an artifact from a complex set of goals, constraints and context is the core concept of *design*. No field understands this better than the discipline of architecture.

Architecture creates complex, contextualized artifacts primarily for human use in the physical world. Design results from the integration of knowledge, skills and values through a system of practice pervaded by intuition and experience.

> Designers put things together and bring new things into being, dealing in the process with many variables and constraints, some initially known and some discovered through designing. Almost always, designers' moves have consequences other than those intended for them. Designers juggle variables, reconcile conflicting values, and maneuver around

> constraints—a process in which, although some design products may be superior to others, there are no unique right answers (Schön, 1987, p. 42).

The process of design is characterized by an iterative application of understanding, abstracting, structuring, representing and detailing rather than a systematic top-down method (Crampton-Smith & Tabor, 1996). Indeed, part of the process of design is the discovery of what the problems are to be solved. Furthermore, design solutions are characterized by tradeoffs rather than distinct correct solutions as might be the case in a problem-solving approach.

Thus, design is entirely contingent upon judgment—judgment initiates and ends the design process by determining both where to begin and where to stop; it establishes whether or not the design is an appropriate fit with the specific task at hand; and it balances between competing and conflicting values such as environmental, social, economic, and aesthetic factors (Archer & Roberts, 1979; Layton, 1993). The intrinsic aim of design education is "to provide a progression of design activities that enable students to learn to make the value judgments that they must reach in order for designing to proceed" (Norman, 1998, p. 78).

Two trends--the increasingly complex nature of software itself and the need for improved usability--have required moving beyond regarding computer science as simply a type of engineering into exploring the nature of computer science as a design discipline. Due to the explosion in the 1990s of interactive technologies such as Web user interfaces and mobile communication networks, most software artifacts must now integrate network services, and computer and communications hardware into a complex context of software operating system. With the development of these highly human interactive computer consumer products, the nature of software artifacts has changed. Today, most software artifacts are designed for human use.

Because of the rigorous demands of human use, one of the first areas of computer science to explore teaching methods that are typical within architectural design has been the area of HCI, which focuses on the production of interactive artifacts and their end-user interfaces. Examples of this are far-ranging, and include well known applications such as the graphical user interface (GUI) desktop for managing files and applications, standard productivity software, email applications, social networking sites, etc. HCI links key concepts of human behavior and design to provide a platform for assessing the usability of everyday objects. Its commonly used methods include User-Centered Design (UCD), Participatory Design (PD), requirements analysis, prototype implementation and evaluation.

Given the unique and difficult nature of architectural design, schools of architecture have evolved a culture of learning tailored to teaching design. The first years in the curriculum focus on the learner, and emphasize inquiry-based approaches and the discernment of fundamental principles. A number of teaching areas build upon this foundation as the students engage increasingly complex, context-rich design tasks,

including representation, case studies of design, and analytic tools and techniques for determining feasibility and results (for example, energy loss from site specific windows or load factors for walls). Many of these approaches may also be appropriate methods of teaching other design disciplines and are worthy of investigation. Most significant to this study is the unique place where all of this is brought to hand--the studio.

## REPORTS ON THE STUDIO

Although ascribed as the common core of design education, the studio is often taken for granted by architecture and design faculty who were themselves educated in studios. In the last decade two significant reports refocused attention on the studio as a unique and exemplary setting for learning.

The 1998 Boyer report *Reinventing Undergraduate Education* brought attention to the "scruffy architecture studio" as a model environment supporting learning and discovery, as substantiated by recent transfers of the studio method to other disciplines such as Richard Fould's *New Jersey Institute of Technology* (NJIT) introductory biomedical engineering course (DeLoughry, 1995; Foulds and Bergen, 2001; Loss and Thornton,1998; Wilson & Jennings, 2000). The studio format is commonly implemented in non-design disciplines through problem-based learning (PBL), a change to the curriculum that combines context-rich open-ended problems with collaborative learning. The role of the faculty switches from content delivery agents (Dinham's "controller of information") to on-demand consultants ("orchestrator"), displacing them from the front of the classroom to the back, where they rove amongst students working in groups. Foulds' format is typical given the constraints and objectives of teaching in disciplines where students are taking 4 to 5 classes of equivalent weight in a semester, rather than a design studio of 6 to 9 credits. He structures each class session around well-defined learning objectives, planning time frames for mini-lectures, active learning tasks, and a wrap-up discussion at the end (Foulds, p. 95).

In 2000 the AIAS (American Institute of Architecture Students) *Studio Culture Task Force* brought a brief, feverish pitch of attention to the tacit social underpinnings of studio. Manifest at its worst, the architecture students in the AIAS task force perceived studio culture as indoctrination or hazing. In contrast, for Brown (2006), the "enculturating" process *is* what distinguishes the studio as a learning environment. Brown looks towards open expert communities on the internet as examples of "learning to be-" through what he terms peripheral participation-apprenticeships that model:

> A way of seeing
> A way of knowing
> Sensing what constitutes an interesting problem
> Knowing what constitutes an elegant solution
> Being able to engage in productive inquiry [productive inquiry is that aspect of any activity where we are deliberately (though not always consciously) seeking what we need, in order to do what we want to do –e.g. leveraging

the net] (Brown, 2006, p. 19).

TEACHING DESIGN: *An ecology of learning*

As we looked at how people from other disciplines were deploying studio, we realized that at the heart of our investigation was an essential question about the nature of studio, and what contributes to the studio's success in cultivating learning, in particular, in learning how to design. Therefore it is critical to distinguish that in applications lauded by the Boyer Report, "studio" is typically defined as a sub-category of active learning, and not necessarily bound to design learning. In design epistemology, the "products" of learning are in fact the designs themselves. That is actually the knowledge that is learned in the studio: how to create and communicate a design--or better--a good design. The literature provides evidence that the studio techniques used in architecture education have been used effectively to prepare designers in many other disciplines, including industrial design, engineering, the physical sciences, and computer science (Gottfried, et. Al, 2007; Little & Cardenas, 2001; Reimer & Douglas, 2003).

Within the context of this study our initial deliberations began with an inventory of the similarities and differences between the typical learning environments of design schools and computer science departments. These discussions brought together two points of view: one, of design faculty (architecture & industrial design from Virginia Tech) who spend 12 to 16 hours a week teaching in a studio setting; the other, of outside observers of the studio including the computer science and learning technology educators on the team.

The designers presented studio as a place supporting a unique set of interactions between people, tools, materials and media. The goal of these interactions is to seek understanding through direct inquiry in order to carry ideas into the concrete world of experience and utility.

Reimer and Douglas, the computer scientists on the research team, used these characteristics and their previous study of architecture studios at the University of Oregon to frame our conversation about the nature of studio (Figure 1):

Supervision by a master designer
Studio space**
Studio time**
Design problem
Periodic lectures**
Student collaboration
Critiques (including desk crits by instructors, pin-ups, interim crits, final crits)**
Exit interviews
** Where we found major differences between the design cultures

**Figure 1. How do different disciplines (Architecture, Industrial Design, Computer Science) consider the elements of what makes a studio**?

As an initial framework for assembling a cross-disciplinary understanding of studio, we adopted Shaffer's (2007) definition of studio which was based on his observations of an architecture studio at MIT. Shaffer defines the studio as a "coherent system" in which surface structure, pedagogy and epistemology *interact* to create a unique learning environment. Shaffer's 2007 study of "what makes a studio" included the following essential factors:

- **Surface structure** refers to the logistics of the studio such as the setting, space, time block, and social context.

- **Pedagogical activities** include the character and duration of assignments, activities and interactions, such as iterative cycles of design, hands-on investigations, and group discussions of work in progress.

- **Epistemological understanding** describes the beliefs about the nature of knowledge and how it is constructed that guide studio activities.

There are obvious distinctions between the learning environments of a design school and those of computer science. Building upon Shaffer's system of "what makes a studio," (Figure 2) our research seeks to identify what amongst these differences--time, physical space, prerequisites (knowledge-based verses meta-cognitive), and warrants--are significant to building a disciplinary *and* cross-disciplinary understanding of the studio.

| | | **Architecture & Industrial Design** | **Computer Science** (HCI in particular) |
|---|---|---|---|
| **STRUCTURAL** | Space | •24/7/365 dedicated space<br>•Designated or borrowed space for pin-ups<br>•Gallery areas | •No dedicated space for students<br>•Standard computer labs or classrooms for crit sessions<br>•No pin-up space |
| | Time | •Class meets 12-16 hours a week (4 hour blocks)<br>•Students work off-hours in shared space | •Class meets 3 hours per week (50 or 80 minute blocks)<br>•Team meetings out-of-class hard to schedule |
| **PEDAGOGICAL** | Lectures | •As-needed talks are a supplement<br>•Students have an introduction course in essential design principles<br>•Balance tilts towards building general tacit knowledge over specific content areas | •Lectures are key component<br>•Students often take only one class in user interface design, so no prior knowledge assumed.<br>•Students must learn content areas |
| | Critiques | •Design crits are most often individual presentations<br>•Presentations occur frequently while the work is in-progress and with varying degree of formality<br>•Feedback given by classmates, instructor and guests (practitioners or educators) | •Design crits are oral presentations by team<br>•Feedback given by classmates and instructor |
| **EPISTEMOLOGICAL** | Assessment | •Grades are given for the entire semester's work, not for individual projects.<br>•Based on:<br>--over all quality of work<br>--progressive body of work… the student has shown a trajectory of growth in their projects from beginning to end...no plateau, or up and down, but a consistent striving to do better.<br>--intellectual development of student | • Students receive grade for each project or milestone<br>•Individual grade often equals team grade<br>•Mid-term written exams typical to test "content"<br>•Extra work or projects for graduate students in class |

**Figure 2.** ***An ecology of learning, not a taxonomy.*** Building upon Shaffer's system of "what makes a studio."

## HCI AS A DESIGN DISCIPLINE: Current Work

We are in the process of collecting data from a User Interface (UI) design course that uses a "hybrid" studio model currently being taught at the University of Montana. Our goal is to analyze a variety of design process artifacts from this class to help us refine a studio-based curriculum for teaching design in HCI. The remainder of this paper will describe the course format, projects and challenges we have faced so far this semester.

There are four graduate Computer Science (CS) students enrolled in this class and 5 undergraduate CS majors. The course meets two days a week, 80 minutes at a time, for a total of 15 weeks. The students were told on the first day of class that we would be collecting data and artifacts on the process of design throughout the semester, and they were given the chance to opt out of any of the collection methods. The data collection methods we are using this semester include videotaping design critique (or pin-up) sessions, select group work meetings, and instructor desk crits (critiques). We are also collecting copies of design artifacts including low and high-fidelity prototypes, written reports, design journals, peer evaluations, BlackBoard discussion threads, instructor reflections documentation, and end-of-semester questionnaires.

The course consists primarily of lecture periods, but has been structured to allow for design pin-ups each time a major project assignment is due, and two in-class desk crit sessions. The content areas covered during lecture, which are considered core UI design concepts that students must understand, include: Introduction to human behavior, interaction, and usability; Principles of good design; and the UCD methodology, including requirements analysis, preliminary design, iterative development, low and high-fidelity prototyping, heuristic evaluation, cognitive walkthrough, and usability testing.

To ensure that students are grasping the key concepts of UI design covered in lecture, there is one midterm exam scheduled. To provide opportunities for students to apply these concepts to practical application design, there are five major assignments given to teams of students (generally 2-3 per team) throughout the semester. Each time a project is due, with the exception of the final project, which is due during the two-hour final exam period, a full week (two 80 minutes classes) is allocated to in-class design critique sessions. Teams are given approximately 30 minutes to present their work to the rest of the class for feedback and suggestions. Teams use a LCD projector to display their work on a whiteboard situated in the front of the classroom.

This semester, for projects 1 & 2, students selected an existing product from a well-known technology and gadget catalog, and were asked to re-design it. Project 1 focused on the physical aspects of the product along with approximately three functional software aspects. Project 2 required students to iterate on their initial design by incorporating suggested changes received during design crits and via instructor review, to extend the scope of the functionality included in their designs, and to conduct a heuristic evaluation

(Lewis & Rieman, 1994) on their preliminary designs. Three of the four teams chose a digital picture frame for their design challenge, while the fourth team chose a hand-held golfing GPS device. The deliverables for both project 1 & 2 were low to medium-fidelity prototypes in the form of hand drawn sketches or computer generated diagrams, and a written report.

Projects 3, 4, and 5 involve having student teams create their own interactive software system and then design (Project 3), prototype (Project 4), and evaluate this system with actual or potential end users (Project 5). The primary deliverables for Project 3 include low to medium-fidelity prototypes (sketches, screen mock-ups, computer generated drawings) and a written report that describes the targeted user population for the system and the functional requirements. For Project 4, students are expected to incorporate design feedback into the next version of their design and create a prototype robust enough for usability testing with end users. Students are allowed to use any programming environment they wish to complete this project, but no class time is dedicated to UI programming techniques. Finally, for Project 5 students conduct usability tests with 4-6 end users. This project includes generating a test plan and necessary testing materials (scripts, scenarios of use, task lists, etc.), achieving Institutional Review Board (IRB) certification necessary for human subjects research, and videotaping testing sessions for subsequent review and analysis.

## CHALLENGES AND QUESTIONS

Some challenges associated with integrating the studio method of teaching design into a standard Computer Science curriculum are previously documented (Reimer & Douglas, 2003). The experiences this semester reaffirm most of those challenges, and bring new issues and questions to the forefront. For example, we understand that the physical space typically allocated to students differs dramatically between Architecture or Industrial Design departments and Computer Science departments. However, this semester as we try to make more room in the schedule for in-class desk crits, we realize that the problem often extends beyond dedicated worktables for students and wall space for design pin-ups. In particular, the course is currently being taught in a Computer Science lab with desktop computers, which makes it very difficult to find the necessary space for teams or groups of students to work together on paper designs (see Figure 3). Unless design specifications are actually on the computer, which is rare in early design, our need for open tables and increased desk space conducive to “spreading out” has driven us to a different classroom on desk crit days.

**Figure 3. Space problems for design team.**

Another challenge we have found in integrating the typical studio method into this course is balancing the need to teach students about content areas central to UI design (and HCI in a broader sense), such as those listed above, and the time necessary for students to engage in the process of iterative design and develop complex software interfaces. This issue is exacerbated by the fact that most CS students only take one course in HCI/UI design--thus cannot rely on building upon previously acquired foundational knowledge in these areas--and that programming a high-level prototype is a time consuming and complex process. One solution might be to eliminate the programming aspect altogether from the course and focus exclusively on designing low-fidelity user interfaces, but then students would not be able to experience conducting interactive usability tests with end users.

Finally, another issue brought to the forefront this semester is the difficulty students have illustrating an interactive design, where the system changes based on user input, in the static medium of low-fidelity prototyping. Storyboarding, where screen sketches or mock-ups are displayed one after another with written narration, is a method commonly used to show changes in system state. However, this technique can require a significant number of renderings with only minute differences between them, can be tedious and time-consuming, and is often simply impractical for showcasing large portions of a system's functionality. Another method sometimes used is program flow diagramming, like that seen in Figure 4, which is an example of student work from this semester. These kinds of diagrams are particularly useful for showing the overall scope of a system and how user input can result in different system states, but it may be difficult to illustrate particular screen details and the depth of certain system functions.

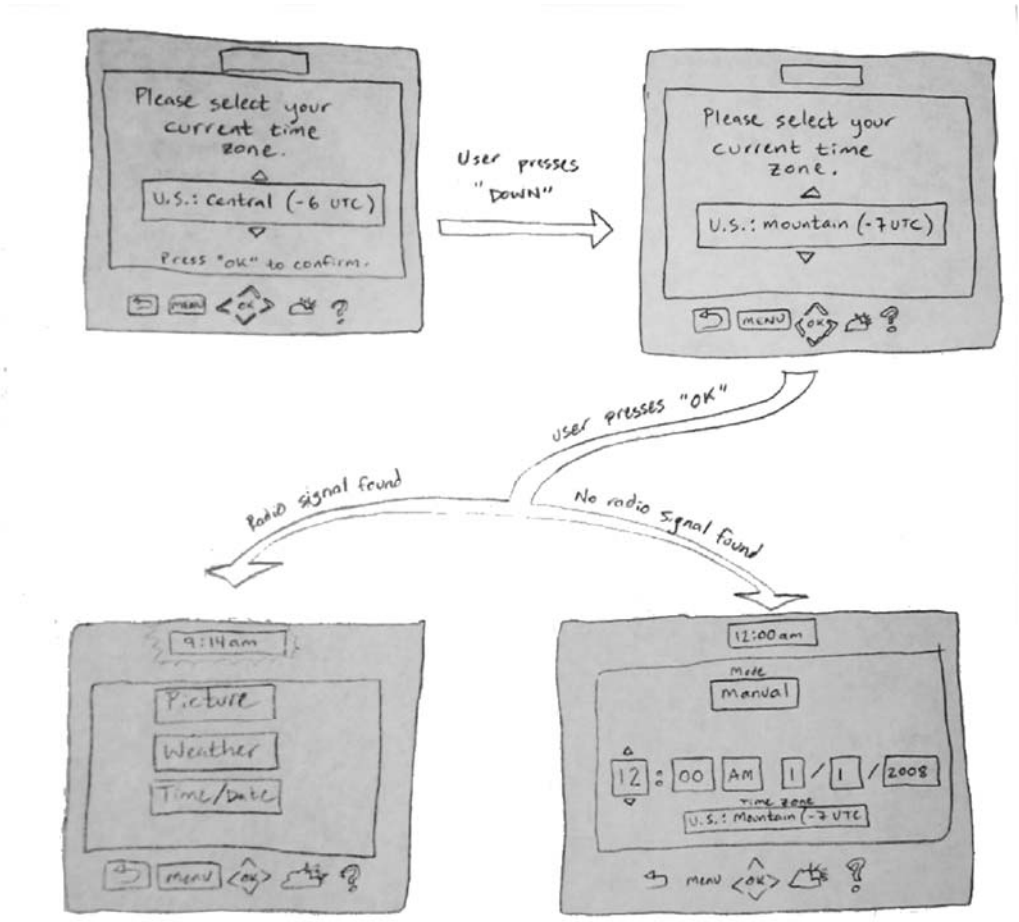


**Figure 4. Example of Student work, program flow diagram.**

We understand that some of the issues described are unique to software design, in particular user interface design, and our challenge is to determine how to successfully navigate these problems within the broader scope of using the studio method. We believe that the benefits of using the studio method are significant, and we are confident that there are effective ways of incorporating it into a typical CS curriculum. Despite the challenges we face, there are also numerous success stories that we will enumerate and describe further upon analysis of the data we are currently gathering.

## DISCUSSION: Questions to Take Back to Architectural Design Education

Many studies have been done concerning what constitutes effective design studio methodology, although very little of the research carried out can be verified statistically through empirical data. Part of the explanation for the scant number of empirical studies can be explained by the difficult and complicated nature of social sciences research and the evolving and controversial nature of design itself. A larger volume of research on the topic is available from case studies and other qualitative research. Based on our preliminary investigation, we have generated as many questions as answers.

Research on the cross-disciplinary transfer of studio education, in particular the transfer

to computer science, holds a potential to generate lessons that can also inform the nature of studio education at its source in the design disciplines. The Bauhaus legacy of studio has remained largely intact for three-quarters of a century, all the while the profession of architecture has experienced significant innovations in tools and forms of practice. This leads us to ask if there are aspects of the traditional studio that have become obsolete, a question that we plan to address through identifying interactions within the study and reassessing through qualitative analysis their contributions to the studio's success in cultivating learning how to design in today's technology-laden environment.

Aspects of all three of Shaffer's interactions--structure, pedagogy and epistemology—are at odds with normative university practices. Allocations of space, time, and faculty and teaching methods prove to be among the most challenging elements to transplant into the wider university. Until very recently, in most schools of design these very elements--time (the dedication of significant time blocks in the curriculum and continuous access to a dedicated space) and distinct pedagogical practices--were considered quintessential to the definition of studio itself. New technologies such as wikis and high-band video connections are fostering experiments in "virtual studios," (Andia, 2002; Blevis, Lim, Stolterman & Makice, 2008; Brown & Cruickshank, 2003) and online studio education, both as a form of distance learning and as a "hybrid" complement to traditional face-to-face interactions (Bender, 2005; Dave & Danahy, 2000; Kvan, 2001; Notash & Garriock, 2003; Seitamaa-Hakkarainen, Lahti, & Hakkarainen, 2005; Zimring, Kahn, Craig, Haq, & Guzdial, 2001). It is too early to confirm the impact on the communal aspects of studio learning that ensue from giving up a dedicated physical space and face-to-face interactions. Nevertheless, these studies reveal a portion of the transformative impact of technology on studio education in the emerging environments brought about by technology (including the segue from physical to virtual desktops, and the use of iPods and wireless communication).

In computer science, as with many of the other disciplines experimenting with the studio pedagogy, there is a perceived epistemological gulf between a discrete, knowledge-based disciplinary education and the often indeterminate knowledge of design. Indeed, many in the architectural community view architecture as "an information-deprived profession" (Spreckelmeyer, 1993) and have voiced a concern to increase the emphasis on content-knowledge in the studio. Often the necessity to deliver content-based knowledge results in a hybrid studio--a mix between lecture and studio components where the faculty switches from delivery agent to mentor within one time block or course module. Is that possible? Can we ask our students to be consumers and constructivists of knowledge within the span of a class meeting? The integration of applied science and technologies not only within the studio environment, but also between studio and the rest of the curriculum, has been struggle many architectural educators have faced since the inception of architecture schools (Allen, 1997; Brady, 1996; Levy, 1980; Norman, 1998; Schön, 1998).

## SUMMARY

In summary, this paper recounts the clarifications, challenges and questions resulting from a cross-disciplinary dialogue about studio education that ran in tandem with our first

year of data collection in a 3-year collaborative research and education project. The first two years of the project will focus on research involving data collection and analysis that will lead to the development curriculum guidelines for use in HCI instruction. Through an investigation of the design process used within an interdisciplinary studio-based project and HCI courses that incorporate a modified studio method, we will derive principles that can be applied to the education of future computer science professionals. Through this process, we hope to illuminate methods that might be effective in teaching architecture and industrial design students as well.

ACKNOWLEDGEMENTS

The authors wish to acknowledge the contributions Miriam Larson who conducted the literature review on studio education for the project.

This material is based upon work supported by the National Science Foundation (NSF) Computer and Information Science and Engineering (CISE) Science of Design under Grant No. 0725290, *Collaborative Research: Investigating and Refining the Studio Experience as a Method for Teaching Human Computer Interaction*. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

BIBLIOGRAPHY

Allen, E. (1997). Second Studio: A Model for Technical Teaching. *Journal of Architectural Education, 51* ( 2 ), 92-95.

Andia, A. (2002). Internet Studios: Teaching Architectural Design On-Line between the United States and Latin America. *Leonardo, 35*( 3), 297-302.

Archer, B. & Roberts, P. H. (1979). Design and Technological Awareness in Education. *Studies in Design Education Craft & Technology, 12*(1), 55–56.

Bender, D. (2005). Developing a Collaborative Multidisciplinary Online Design Course. *The Journal of Educators Online*, *2*(2), 1-12.

Blevis, E., Lim, Y,-K., Stolterman, E, & Makice, K. (2008) The Iterative Design of a Virtual Design Studio: The Instructional Design Portfolio. *TechTrends, 52* (1), 74-83.

Boyer Commission (1998). *Reinventing Undergraduate Education, a Blueprint for America's Research Universities.* Stony Brook, NY: State University of New York.

Brady, D. A. (1996). The Education of an Architect: Continuity and Change. *Journal of Architectural Education, 50*(1), 32-49.

Brown, S., & Cruickshank, I. (2003). The Virtual Studio. *The International Journal of Art & Design Education, 22* (3), 281-288.

Brown, J.S. (2006). New Learning Environments for the 21st Century: Exploring the Edge. *Change: The Magazine of Higher Learning, 38*(5), 18-24.

Crampton-Smith, G. & Tabor, P. (1996). The Role of the Artist-designer. In T. Winograd (Ed.), *Bringing Design to Software*. New York: ACM Press.

Dave, B., & Danahy, J. (2000). Virtual Study Abroad and Exchange studio. *Automation in Construction, 9,* 57-71.

DeLoughry, T.J. (1995, March 31). Studio Classrooms. *Chronicle of Higher Education*, p. A19-21.

Dinham, S.M. (1988). Teaching as Design: Theory, Research, and Implications for Design Teaching. *Design Studies,*

*10*(2), 80-88.

Foulds, R.A. & Bergen, M.T. (2001) Studio-based Learning in Biomedical Engineering. *Annual of Biomedical Engineering*, *29*( S1), 104.

Gottfried, A.C., Sweeder, R.D., Bartolin, J.M., Hessler, J.A., Reynolds, B.P., Stewart, I.C., CopCola, B.P., & Holl, M.M.B. (2007). Design and Implementation of a Studio-based General Chemistry course. *Journal of Chemical Education*, *84*(2), 265-270.

Koch, A., Schwennsen, K., Dutton, T.A., & Smith, D. (2002). *The Redesign of Studio Culture: A Report of the AIAS Studio Culture Task Force.* NY: AIAS. Retrieved February 8, 2008 from: http://www.aias.org/studioculture/studioculturepaper.pdf

Kvan, T. (2001). The Pedagogy of Virtual Design Studios. *Automation in Construction*, *10*, 345-353.

Layton, D. (1993). *Technology's Challenge to Science Education*. Philadelphia: Open University Press.

Levy, Alan. (1980). Total Studio. *Journal of Architectural Education*, *34*(2), 29-32.

Lewis, C., & Rieman, J. (1994). *Task-Centered User Interface Design: A Practical Introduction*. Retrieved April 17, 2008 from: http://bmrc.berkeley.edu/courseware/cs160/fall99/Book/

Loss, R. & Thornton, D. (1998). Physics Studio - A Progress Report. In Black, B. and Stanley, N. (Eds), *Teaching and Learning in Changing Times, Proceedings of the 7th Annual Teaching Learning Forum*, (pp. 171-175). Perth: University of Western Australia, Retrieved March 30, 2008 from: http://lsn.curtin.edu.au/tlf/tlf1998/loss.html

Little, P. & Cardenas M. (2001). Use of "Studio" Methods in the Introductory Engineering Design Curriculum, *Journal of Engineering Education*, *90*(3), 309-318.

Norman, E. (1998). The Nature of Technology for Design. *Journal of Technology and Design Education*, *8*, 67–87.

Notash, L., & Garriock, M. (2003). International Virtual Design Studio (IVDS) Program: Overview, Challenges, and Students' Perspectives. *IEEE Transactions on Education*, *46*(3), 345-349.

Reimer, Y.J., & Douglas, S.A. (2003). Teaching HCI Design with the Studio Approach. *Computer Science Education*, *13*(3), 191-205.

Schön, D.A. (1987) *Educating the Reflective Practitioner*. San Francisco: Jossey-Bass.

Schön, D. A. Toward a Marriage of Artistry & Applied Science in the Architectural Design Studio. *Journal of Architectural Education*, *41*(4), 4-10.

Seitamaa-Hakkarainen, P., Lahti, H., & Hakkarainen, K. (2005). Three Design Experiments for Computer-supported Collaborative Design. *Art, Design & Communication in Higher Education*, *4*(2), 101-119.

Shaffer, D. W. (2007). Learning in Design. In R. A. Lesh, J. J. Kaput & E. Hamilton (Eds.), *Foundations for the Future In Mathematics Education*, 99-126. Mahweh, NJ: Lawrence Erlbaum Associates.

Spreckelmeyer, K. (1993). An Introduction to the Symposium. *Knowledge-based Architectural Education: Reconfjguring the Studi.* San Antonio, TX: Architectural Research Center Consortium.

Wilson, J. & Jennings, W. (2000). Studio Courses: How Information Technology is Changing the Way We Teach, on Campus and off. *Proceedings of the IEEE*, USA, *88*, 72-80.

Zimring, C., Khan, S., Craig, D., Haq, S., & Guzdial, M. (2001). CoOL Studio: Using Simple Tools to Expand the Discursive Space of the Design Studio. *Automation in Construction*, *10*, 675-685**.**